



Application-Value Billing Server Simulator

User Guide

QUALCOMM Incorporated
5775 Morehouse Drive
San Diego, CA 92121-1714
U.S.A.

Copyright © 2004–2012 QUALCOMM Incorporated.
All Rights Reserved.

Not to be used, copied, reproduced in whole or in part, nor its contents revealed in any manner to others without the express written permission of Qualcomm.

This technical data may be subject to U.S. and international export, re-export or transfer (“export”) laws. Diversion contrary to U.S. and international law is strictly prohibited.

QUALCOMM, Brew, and the Brew logo are registered trademarks of QUALCOMM Incorporated in the United States and may be registered in other countries.

Other product and brand names may be trademarks or registered trademarks of their respective owners.

Application-Value Billing Server Simulator User Guide

80-D4692-1 Rev. F

June 15, 2012

Contents

1 Introduction	4
2 Running the A-VB Server Simulator.....	5
3 Debugging transactions	6
4 Changing adversarial test conditions	9
5 Setting server preferences.....	10

1 Introduction

Brew™ value-billing applications create billable transactions, which are submitted over the operator network to the Brew Distribution System (BDS). During submission, conditions may arise where the Brew application needs to handle unexpected errors ranging from network problems to the subscriber not having enough funds to pay for the transaction. The Application-Value Billing (A-VB) Server Simulator allows the developer to simulate these errors. Developers can also use the A-VB Server Simulator to verify that the intended data is being sent from their Brew application.

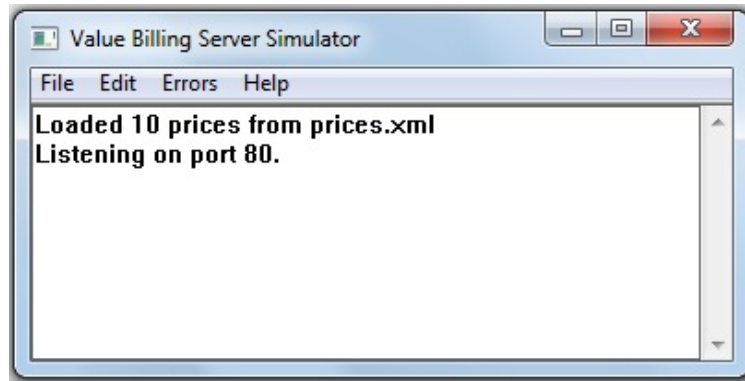
2 Running the A-VB Server Simulator

There are two methods for running the server: from the Windows desktop and from the command line.

To run the A-VB Server Simulator from the Windows desktop

1. Navigate to the folder that contains the BREW_VBillingServer.exe file. The default location is C:\Program Files\BREW x.x.x\sdk\bin.
2. Double-click **BREW_VBillingServer.exe**.

The Application-Value Billing Server Simulator window opens.



Use the menus as follows:

Click this menu	To do this
File	Save messages and exit the server.
Edit	Clear messages and edit preferences.
Errors	Enable and disable adversarial testing conditions.
Help	View version information.

To run the A-VB Server Simulator from the command line

1. At the command prompt, navigate to the folder that contains the BREW_VBillingServer.exe file.
2. Type `BREW_VBillingServer.exe`.
3. Use the menus as described in the previous procedure.

3 Debugging transactions

The server shows information about each transaction it receives.

Figure 1 shows a successful transaction that includes a price, currency, short description, long description (SJIS text), 10 bytes of vendor data, and a payee ID. The result is 0 (success) with a message of: ok.

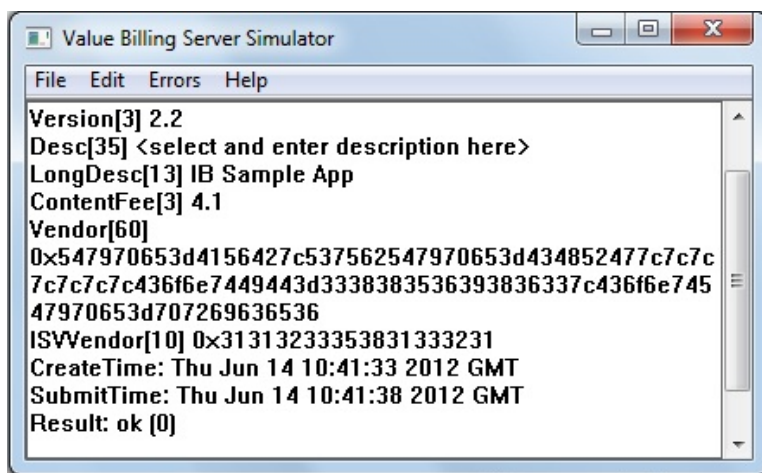


Figure 1. Example: Successful transaction

Figure 2 shows the results of selecting **SID not authorized** from the Errors menu and resending the same transaction as used for Figure 1. Error 3 is returned to the Brew application.

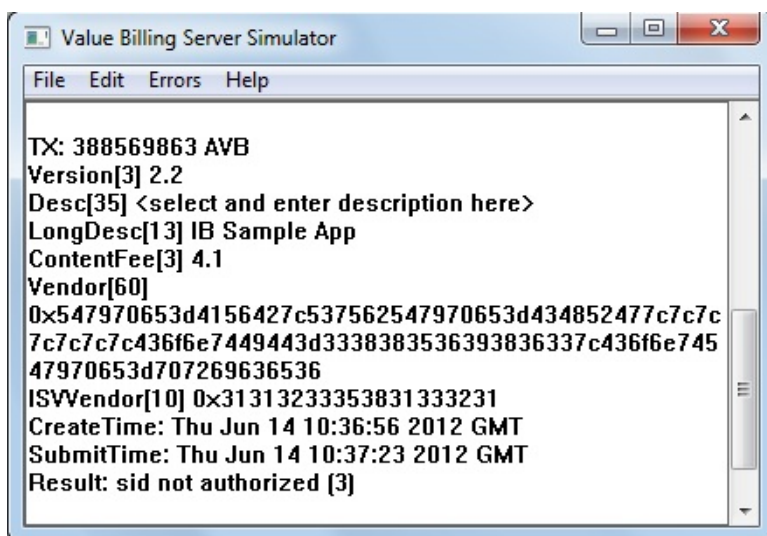


Figure 2. Example: SID not authorized

Figure 3 shows the result of sending a short description that contains characters outside the allowed range (unsupported UTF-8 characters are detected). Error 6 is returned to the Brew application.

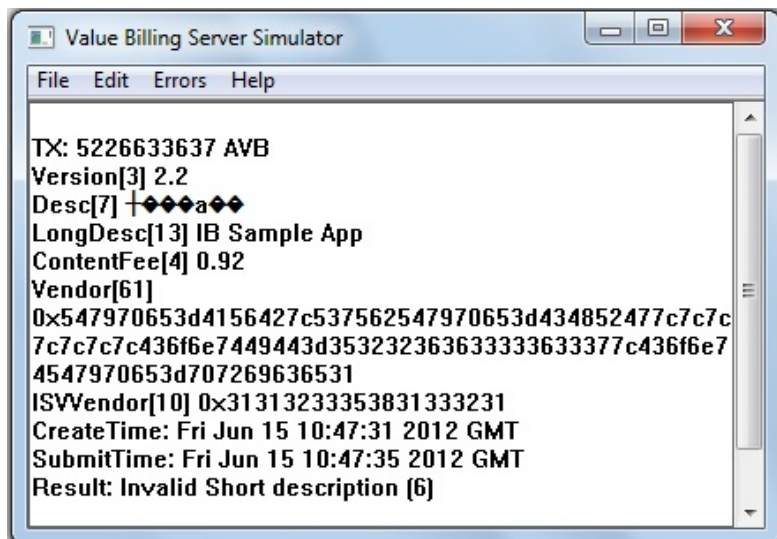


Figure 3. Example: Invalid short description

Figure 4 shows the result of sending vendor data that contains characters outside the allowed range (unsupported UTF-8 characters are detected). Error 7 is returned to the Brew application.

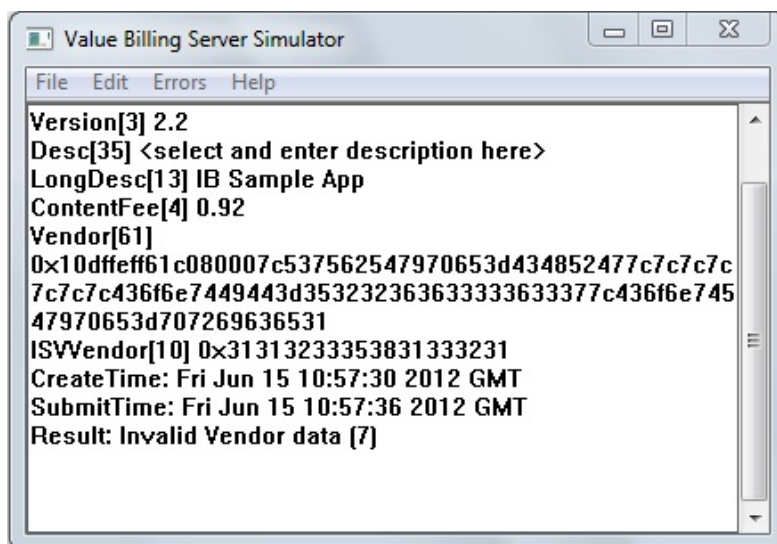


Figure 4. Example: Invalid vendor data

Figure 5 shows the result of sending a long description that contains characters outside the allowed range (unsupported UTF-8 characters are detected). Error 6 is returned to the Brew application.

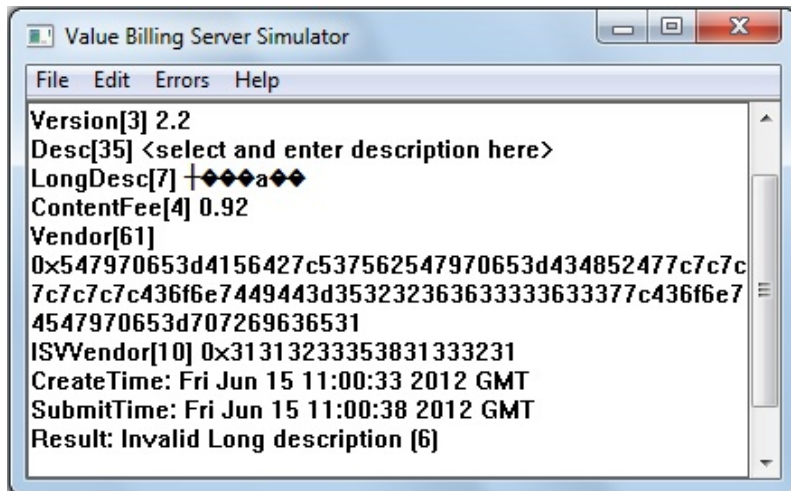


Figure 5. Example: Invalid long description

4 Changing adversarial test conditions

Use the Errors menu to enable and disable adversarial testing conditions.

To change adversarial testing conditions

1. Click **Errors**.
2. Click the condition to use for adversarial testing.

The following conditions are available:

Condition	Description
Invalid Content Fee	The content fee is invalid (OPT_CONTENTFEE).
SID Not Authorized	Authorization of the subscriber failed.
Invalid Price	The price does not exist, is non-numeric, or is less than or equal to zero (OPT_AMOUNT).
Invalid Currency	The currency does not exist or is invalid for the carrier (OPT_CURRENCY).
Invalid Billing Description	The short description does not exist or is invalid (OPT_DESCRIPTION).
Invalid Billing Data	General error indicating that a field such as the vendor data is too large.
Insufficient Funds	A prepaid subscriber does not have enough funds to complete the transaction.
Random Error	Randomly select one of the above error conditions.
HTTP Server Error	Return of an HTTP 500 error.
No Response	The server will not send any response.
Delayed Response	The server will not respond for the number of seconds specified in the Delayed Response preference setting. The default is 15 seconds.

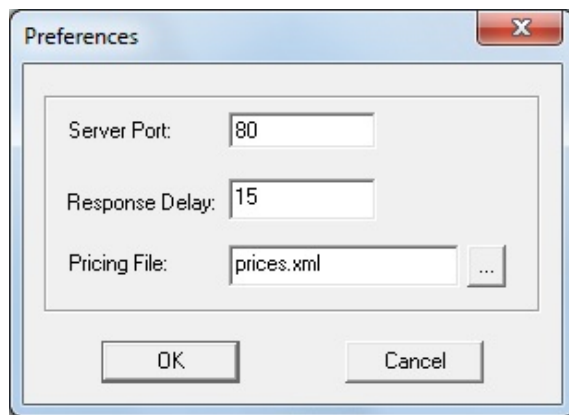
5 Setting server preferences

Using the **Edit > Preferences** dialog box, you can change the following server preferences:

- **Server Port** – By default, the server listens for requests on TCP port 80. This port can be changed in the preferences dialog box. The port must also be changed in the Brew application.
- **Response Delay** – Used with the Delayed Response adversarial test. The default is 15 seconds and can be set to any value greater than zero.
- **Pricing File** – Used to specify the XML file containing all the pricing information.

To change the server settings

1. Click **Edit > Preferences** to show the Preferences dialog box.



2. Enter the server port number.
3. Enter the response delay.
4. Browse to select the pricing file.
5. Click **OK**.

The pricing XML file must adhere to the following XML schema:

```
<?xml version="1.0"?>

<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <xsd:element name="prices" type="PriceList"/>

  <xsd:complexType name="PriceList">
    <xsd:sequence>
      <xsd:element name="price" type="Price" minOccurs="1" maxOccurs="unbounded"/>
    </xsd:sequence>
  </xsd:complexType>

  <xsd:complexType name="Price">
```

```

<xsd:sequence>
  <xsd:element name="id" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="method" type="xsd:unsignedInt" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="basis" type="xsd:unsignedInt" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="basisValue" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="handle" type="xsd:unsignedLong" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="display" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="value" type="xsd:float" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="dap" type="xsd:float" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="currency" type="xsd:string" minOccurs="1" maxOccurs="1"/>
  <xsd:element name="description" type="xsd:string" minOccurs="0" maxOccurs="1"/>
</xsd:sequence>
</xsd:complexType>

</xsd:schema>

```

Following is an example pricing XML:

```

<?xml version="1.0" encoding="utf-8"?>
<prices xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:noNamespaceSchemaLocation="prices.xsd">
  <price>
    <id>price1</id>
    <method>8</method>
    <basis>0</basis>
    <basisValue>0</basisValue>
    <handle>100001</handle>
    <display>$1.00</display>
    <value>1.00</value>
    <dap>0.50</dap>
    <currency>USD</currency>
  </price>
  <price>
    <id>price2</id>
    <method>8</method>
    <basis>0</basis>
    <basisValue>0</basisValue>
    <handle>100002</handle>
    <display>$2.00</display>
    <value>2.00</value>
    <dap>1.00</dap>
    <currency>USD</currency>
  </price>
  <price>
    <id>price3</id>
    <method>8</method>
    <basis>0</basis>
    <basisValue>0</basisValue>
    <handle>100003</handle>
    <display>$3.00</display>
    <value>3.00</value>
    <dap>1.50</dap>
    <currency>USD</currency>
  </price>
</prices>

```